TM-1224
1700

# A COMPUTER PROGRAM FOR CALCULATING THE RESONANT FREQUENCY, SHUNT IMPEDANCE AND QUALITY FACTOR OF A PILL-BOX CAVITY IN A STORAGE RING

Victor M. Aguero

and

King Yuen Ng

(October, 1983)

# I. INTRODUCTION

Keil and Zotter[1] have analyzed the electromagnetic fields excited by the longitudinal density fluctuations of an unbunched relativistic particle beam drifting in a corrugated vacuum chamber of circular cross section as shown in Figure 1. At higher frequencies, these corrugations become resonant cavities. Zotter has written a computer program known as KN7C to compute the resonant frequencies. However, in the actual use of KN7C, some difficulties are encountered:

1. The program has an input parameter D which is the period of the corrugation and the allowed value it can assume is of O(meters). This will be very unrealistic if we have only one resonant cavity inside a storage ring of circumference of O(km).
2. One of the input is H defined as "pitch" in meters. Its meaning is totally unclear.
3. The program only computes the resonant frequencies but tells us nothing about the shunt impedances and the quality factors.
4. The program is very long but carries not a single comment statement; so it is nearly impossible for one to follow. The situation is even more obscure because KN7C has not been fully converted into a version applicable to the Fermilab computer.

Because of the above, we attempt to write instead another computer program known as CAVITY to analyze this pill-box shaped

resonant cavity. Although there are many input variables to this program, only two are essential and need to be specified. They are BD = b/d = the ratio of the circular beampipe radius to that of the pill-box cavity and GD = g/d where g is the length of the cavity. When they are specified, CAVITY will print out the dimensionless normalized fundamental resonant frequency FD, shunt impedance Z and figure of merit Q. From these, the actual resonant frequency, shunt impedance and figure of merit can be deduced:

$$\text{resonant frequency} = FD*c/(2\pi d),$$

$$\text{shunt impedance} = (1-i)1.5Z_o Z(\sigma d)^{1/2},$$

$$\text{quality factor} = Q(\sigma d Z_o)^{1/2},$$

where c is the velocity of light, $Z_o = 120\pi$ ohms is the impedance of free space and $\sigma$ is the conductivity of the curved cavity walls. The two annular ends are assumed to be perfectly conducting. If another input variable MODES = N is specified, the resonant frequencies will be computed from the fundamental mode to the Nth higher mode.

The program, the input and the output will be described in the following sections.

## II. THE PROGRAM CAVITY

### II.1 General description

In analyzing the resonant cavity, we follow exactly the

method given by Reference 1; i.e., solving for the field coefficients from the matrix equation

$$\bar{X} = \alpha(U-\eta_t I)^{-1}N[U-\alpha SN\dagger I(U-\eta_t I)^{-1}N]^{-1}SN\dagger(U-\eta_t)^{-1}\bar{Y},$$

which is Eq. (3.5) of Reference 1, where the matrices and symbols in above are defined. To solve for the resonant frequency, we look for zeroes of the real part of the determinant of

$$W = U-\alpha SN\dagger I(U-\eta_t I)^{-1}N,$$

the dimension of which is truncated to the order $g/\lambda$, where $\lambda$ is some small length inside which the field pattern around the cavity edges changes significantly and should be taken as the smaller of g, b, or d-b. This dimension is denoted by NS in the program and is an input; the default is set at NS=6. If NS is not big enough, a warning statement will be printed suggesting a more suitable value.

For the above matrices, there is another dimension which is related to the harmonic number around the circumference of the storage ring. If a truncation is attempted during a summation over the harmonic number, we need to go up to $O(2\pi R/\lambda)$ where R is the ring radius. However, in every such summation, the elements of the sum are only slowly varying and the summation can therefore be approximated by an integral. In the program, we integrate from 0 to XI with an absolute (or relative) error ERROR and from XI to XMAX with an absolute (or relative) error

10*ERROR. Here XI, XMAX and ERROR are input variables; the defaults are XI=10, XMAX=100 and ERROR=.001.

To search for a resonance frequency f, we increase the quantity FD=2πfd/c from the initial value FD (an input, the default is 2.405) by steps of GAP (an input, the default is .02). If Re(detW) changes sign after a step, a zero is detected. We then go back to the former 2πfd/c and increase it by GAP/e instead (e=2.71828). This iteration continues until either the final step size becomes GAP/exp(MM-1) or the total number of steps taken exceeds MAXSTEP. Both MM and MAXSTEP are input variables with defaults set at MM=15 and MAXSTEP=175. The final step size is therefore the accuracy of 2πfd/c at resonance.

Sometimes Re(detW) changes sign only for a very small region in 2πfd/c. If the step size is too big, two closely spaced resonances will be missed. To remedy this, we check the change in slope of Re(detW) also. Thus, if Re(detW) remains positive (negative) and the slope changes from negative to positive (positive to negative), two closely spaced resonances may be present. However, due to the truncation of the size of the matrices, Re(detW) may exhibit some ripples instead. As a result, we include the criterion that the resonances are true only if the absolute value of Re(detW) at the change of slope is less than EPSILON*$10^{-NS/6}$. EPSILON is an input variable; the default is $10^{-6}$.

After tracking the frequency at resonance, the impedance is computed at 2πfd/c + n/10000 with n=0,1,2,...,(NJ-1)/2. The

quality factor is obtained by fitting the admittance (the reciprocal of the impedance) to a Lorentzian curve and the shunt impedance is the impedance at n=0. Here, NJ is an input variable with default set at 21.

If higher modes are required, enter MODES=N in the input. The program will continue to search for zeroes of Re(detW) until the Nth higher resonance is reached. The default of MODES is 1. When Re(detW) is at a zero, Im(detW) will be at an extremum. The shunt impedance will depend on the peak value of this extremum and the quality factor will depend on the shape. Because the higher resonances are usually closely spaced, Re(detW) and Im(detW) fluctuate tremendously. Errors in the truncation of matrices can lead to mismatch between the zero of Re(detW) and the extremum of Im(detW) making the computation of shunt impedance and quality factor impossible. As a result, for higher resonances, only the resonance frequencies are computed.

The program CAVITY can also compute resonant quantities by gradually changing g/d with b/d fixed. To do this, one needs also to enter as inputs KMKM and GDG (defaults are 1 and 0 respectively). The program will start searching for resonances beginning from g/d = GD, then g/d = GD + GDG, ..., and finally g/d=GD+GDG*(KMKM-1).

## II.2 Input and output

Most of the input varibales have been explained in Section II.1. A summary is given as follows:

BD b/d, ratio of beampipe radius b to cavity radius d.

GD g/d, ratio of cavity length g to cavity radius.

FD initial value of $2\pi fd/c$ used in the search of resonances; default is FD=2.405.

GAP initial step size of $2\pi fd/c$; default is GAP=0.02.

MM number of iterations or number of times the step size is decreased by e. The final step size is therefore GAP/exp(MM-1), (e=2.71828). Default is MM=15.

MODES number of modes to be searched; default is MODES=1.

MAXSTEP maximum number of steps allowed in the search of each resonance; default is MAXSTEP=175.

KMKM number of g/d values to be used by the program; default is KMKM=1.

GDG amount of change of g/d on each of the KMKM passes. Since it is more time-saving to search in the ascending frequency direction, one should start from the largest GD desired and work downward with a negative GDG. The default is GDG=0.

NS truncated dimension of the matrix whose determinant is computed in the search for resonances. NS should be even and of $O(g/\lambda)$, where $\lambda$ is the smaller of g, b and d-b. The default is NS=6.

KJ number of points used in the least square determination of the quality factor for the fundamental mode. KJ should be odd. The default is KJ=21.

XMAX the upper limit of an integration which approximates a summation over the harmonic number along the storage ring. It should be bigger than $O(\pi g/\lambda)$, where $\lambda$ is the smaller of g, b and d-b. The default is XMAX=100.

XI an intermediate limit of the integration which approximates a summation over the harmonic number along the storage ring. It should be of $O(\pi g/\lambda)$, where $\lambda$ is the smaller of g, b and d-b. The default is XI=10.

ERROR the absolute error (or relative error whichever bigger) in the integration from 0 to XI, and 10*ERROR is the absolute error (or relative error whichever bigger) in the integration from XI to XMAX. The default is ERROR=.001.

EPSILON EPSILON*$10^{-NS/6}$ is the value which the absolute value of Re(detW) must not exceed to pass the test of a resonance. The default is EPSILON=$10^{-6}$.

GAMMA total energy of the beam particle in terms of the rest energy. The computed results are almost independent of GAMMA when it is bigger than 10. The default is GAMMA=1000.

AB ratio of the beam radius to the beampipe radius. The computed results are insensitive to this variable when the square of $(2\pi fd/c)$*BD*AB/GAMMA is very much less than unity. The default is set at

AB=.005.

GG if set at 'Y', the frequency, determinant, gap size, and slope will be printed out on each step and interation of the search. Any other value will suppress such output. The default is GG=' '.

GQ if set at 'Y', the shunt impedance and quality factor for the fundamental mode of each g/d will be computed. Any other value will suppress such computation. This allows a search to begin above the fundamental mode should higher modes be desired without computing the lower ones first. Mode labels will however be incorrect. The default is GQ='Y'.

The input is in NAMELIST form with the name DATA. Thus, only the entry of BD and GD is deemed necessary because defaults have been assigned for all the rest. An example of an input is

```
$DATA BD=.5,GD=.8,MODES=3$
```

with the first $ sign starting at the second column since the first column of every input record is ignored by the NAMELIST READ statement. The output is shown in Table 1. Several input records can be stacked together and the program will stop and exit after reading an empty input record. Another example for the input is

```
$DATA BD=.5,GD=1.,KMKM=8,GDG=-.1$
```

for search the fundamental resonances when BD=.5 and GD=1.0, 0.9, ..., 0.3. The output is shown in Table 2. Table 3 is an output showing the real part and the slope of the determinant as well as the gap size in each step of the search of a fundamental resonance. The input used is

$$\$DATA\ BD=.5,GD=.8,GG='Y',G_{AP}=.1\$$$

The fundamental resonant frequencies, the shunt impedances and the quality factors (all normalized and dimensionless) have been computed for different values of b/d and g/d (or BD and GD). The results are plotted in Figures 2, 3 and 4. They are essentially the same as those presented in Reference 1. The values of the shunt impedances for b/d=0 are obtained from the known formula for a closed cylindrical cavity:

$$Z = Z_o(c/2\pi fd)(g/\delta)T^2/\pi J_1^2,$$

where $\delta$ is the skin depth of the cavity wall, $J_1$ the Bessel function of order 1 evaluated at 2.405 and T the transit time factor of the beam particle with velocity v,

$$T = \sin(\pi fg/v)/(\pi fg/v).$$

In this limit, the resonance frequency is of course $2\pi fd/c=2.405$ and the quality factor is $(2.405*.5*\sigma dZ_o)^{1/2}$ or $1.097(\sigma dZ_o)^{1/2}$.

The program CAVITY is listed in APPENDIX I for reference.

II.3 How to get and run CAVITY

CAVITY is written using ANSI '77 FORTRAN in CDC FORTRAN 5. The program can be readily obtained from a Fermilab terminal by entering

```
GET,CAVITY/UN=94786
```

It can then be compiled and run by entering

```
R,*
FTN5,I=CAVITY,L
ATTACH,CERNLIB/UN=LIBR
LIBRARY(CERNLIB)
LGO,LFN1,LFN2
```

where LFN1 is the input file in NAMELIST form and the results are written onto the output file LFN2.

CAVITY is also available in the compiled binary form called CAV. Thus, instead of the above, one can also enter

```
ATTACH,CAV/UN=94786
ATTACH,CERNLIB/UN=LIBR
LIBRARY(CERNLIB)
CAV,LFN1,LFN2
```

Because the program solves a matric equation, one can only do it interactively if only a few resonances are searched (in order not to be time-terminated). For more computation, a batch job should be submitted, for example, through the batch job deck

TESTCAV listed in APPENDIX II.

## III. GRAPHIC DISPLAY OF Re(detW) and Im(detW)

As mentioned in Section II.1, the truncation of matrices can lead to error ripples in the determinant. In order to avoid picking up false resonances and have an idea where new resonances will show up, a program CAVPLT has been written to display the behaviors of the determinant. The documentation is as follows:

CAVPLT is a program identical to CAVITY in terms of how frequency and determinant values are calculated. This routine will only do calculations for one set of b/d and g/d values, producing a local file DIGFP (digfile) with which the data are plotted. In addition, a line is plotted along the determinant zero line. A digfile is a graphics display output file developed using the graphics software referred to as DIGS (Device Independent Graphics System).

The input is in the NAMELIST form with the name PLOT. A summary is given below.

BD — same as in CAVITY; no default given.

GD — same as in CAVITY; no default given.

FDST — start of $2\pi fd/c$ in plot; the default is 2.405.

FDEND — approximate end of $2\pi fd/c$ in plot; no default given. GAP step size of $2\pi fd/c$ in making the plot; no default given.

GG — if set at 'R', the plot will be for Re(detW). Any other value will lead to a plot for Im(detW). The default is GG='R'.

NS same as in CAVITY; default is NS=6.

XI same as in CAVITY; default is XI=10.

XMAX same as in CAVITY; default is XMAX=100.

ERROR same as in CAVITY; default is ERROR=.001.

The program CAVPLT is listed in APPENDIX III. It can be readily obtained and run from a Fermilab terminal by entering:

```
GET,CAVPLT/UN=94786
R,*
FTN5,I=CAVPLT,L
ATTACH,CERNLIB/UN=LIBR
ATTACH,GRALIB/UN=LIBR
LIBRARY(CERNLIB,GRALIB)
LGO,LFN,,DIGFP
```

where LFN is the data file specifying the desired plot. The compiled binary version of CAVPLT called CAVP is also available from UN=94786. If too many points are calculated, in order to avoid time-termination, a batch job should be submitted, for example, using the batch job deck TESTP listed in APPENDIX IV. There, the datafile is LFN and the resulting digfile is saved as DIGFP. Any error conditions are reported in a direct access file OUTP, and the dayfile is output to an indirect access file DAYFILP as well as OUTP. (Note that the user identification information must be updated.)

The digfile obtained can be plotted on the CalComp plotters by using the following sequence of commands:

```
GET,DIGFP
GET,QPLOTR/UN=LIBR
QPLOTR,DIGF
```

The output is automatically scaled and rotated for output on 8 1/2 by 11 paper. If one chooses, the digfile can also be viewed directly on a graphics terminal by the commands:

```
GET,DIGFP
ATTACH,PREVU/UN=LIBR
PREVU,DIGFP
```

At the first prompt, enter the transmission rate (120, for example), at the second prompt, enter the reduction factor (.65, for example), at the third prompt, the command PLOT, and then clear the screen for the plot.

As an illustration, a plot of Re(detW) is given in Figure 5 for the input datafile LFN:

```
$PLOT BD=.4,GD=.2,FDST=2.6,FDEND=8.5,GAP=.02$
```

Within the range plotted, there are 6 resonances. Also some ripples are seen.

REFERENCE

[1]E. Keil and B. Zotter, Particle Accelerators 3, 11 (1972).

FIGURE CAPTIONS

Figure 1. Schematic cross section of the corrugated beampipe.

Figure 2. Fundamental resonance frequency versus b/d and g/d.

Figure 3. Shunt impedance at fundamental resonance versus b/d and g/d.

Figure 4. Quality factor at fundamental resonance versus b/d and g/d.

Figure 5. Plot of Re(detW) versus $2\pi fd/c$ for b/d=0.4 and g/d=0.2.

TABLE CAPTIONS

Table 1. Output for $DATA BD=.5,GD=.8,MODES=3$.

Table 2. Output for $DATA BD=.5,GD=1.,KMKM=8,GDG=-.1$.

Table 3. Output for $DATA BD=.5,GD=.8,GG='Y',GAP=.1$.

APPENDIX I

The following is a listing of the program CAVITY written using ANSI '77 FORTRAN in CDC FORTRAN 5. It is available from a Fermilab terminal by entering GET,CAVITY/UN=94786.

```
      PROGRAM CAVITY(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      EXTERNAL F
      COMMON PI,IS,JS,S1,S2,FDC,BD,GD,AB,CNE
      COMPLEX A,X1,X2,XM,F,SUM,Z,UNI,DET,RSS
      COMPLEX CNE,SS,CGAUSS,W(20,20),WW(20)
      CHARACTER GG*1,GQ*1
      DIMENSION V(20),ZR(31),WV(31)
      NAMELIST/DATA/KMKM,MM,BD,GD,GDG,AB,GAMMA,FD,GAP,MODES,
     *              GG,GQ,XI,XMAX,NS,KJ,MAXSTEP,EPSILON,ERROR
      WRITE(6,9999)
9999  FORMAT(1H1)
      WRITE(6,999)
999   FORMAT(" NOTE:       RESONANCE FREQUENCY F IS OBTAINED FROM",
     *       " FD/C BY USING:"//
     *  "                                FD/C = 2*PI*F*D/C,"//
     *  "        WHERE C IS THE VELOCITY OF LIGHT",
     *          " AND D IS THE CAVITY RADIUS."//
     *  "            THE TRUE RESONANCE RESISTANCE IS"//
     *  "                          Z*SQRT(SIGMA*D)*Z0**1.5,"//
     *  "        WHERE Z0 = 120*PI OHMS AND SIGMA",
     *          " IS THE WALL CONDUCTIVITY."//
     *  "            THE FIGURE OF MERIT IS"//
     *  "                            Q*SQRT(SIGMA*D*Z0)."/)
C
C     INITIALIZATION OF VALUES AND DATA INPUT
C
      FCFLAG=0
      PI=3.1415927
      E=2.71828
      XI=10.
      XMAX=100.
      NS=6
      KJ=21
      ERROR=.001
      GG=' '
      MM=15
      GAP=.02
      FD=2.405
      GQ='Y'
      MODES=1
      GAMMA=1000.
      AB=0.005
      KMKM=1
      GDG=0.
      MAXSTEP=175
      EPSILON=1.E-6
9     READ(5,DATA,END=7)
      K=0
C
C     CHECK FOR VALIDITY OF INPUT VALUES
```

```
C
      IF(KJ.GT.31)GO TO 732
      IF(KMKM.LT.1)GO TO 892
      IF(GD+(FLOAT(KMKM-1)*GDG).LE.0.)GO TO 892
      IF(GDG.EQ.0..AND.KMKM.NE.1)GO TO 892
      IF(MM.LT.1)GO TO 892
      IF((BD.LE.0.).OR.(BD.GT.1.))GO TO 892
      IF(GD.LE.0.)GO TO 892
      IF((AB.LE.0.).OR.(AB.GT.1.))GO TO 892
      IF(GAMMA.LE.0.)GO TO 892
      IF(FD.LT.2.40)GO TO 892
      IF((GAP.LE.0.).OR.(GAP.GT.1.))GO TO 892
      IF(MODES.GE.1)GO TO 10
892   WRITE(6,893)
893   FORMAT("//* * * ERROR IN INPUT PARAMETERS * * *//")
      GO TO 9
732   WRITE(6,733)
733   FORMAT("//* * * KJ CANNOT BE BIGGER THAN 31 * * *//")
      GO TO 9
10    WRITE(6,200)BD,AB,GAMMA,NS,XMAX
200   FORMAT(////" B/D = ",F5.4,"   A/B = ",F7.6,3X,
     *"GAMMA = ",F9.4/13X,"NS = ",I3,3X,"XMAX = ",F6.0//)
      X1=(0.,0.)
C
C     DEFINE LIMITS OF INTEGRATION AND CONVERGENCE ERROR ALLOWABLE
C     IN CASES WHERE CONVERGENCE IS BASED ON SLOPE CHANGES
C
      X2=CMPLX(XI,0.)
      XM=CMPLX(XMAX,0.)
      EPSIL=EPSILON*10**(-NS/6)
      FDCPGD=FD
C
C     SET RES. FREQ. OF PREVIOUS G/D RUN AND LOOP THROUGH G/D VALUES
C
      DO 25 KMM=1,KMKM
C
C     CHECKING WHETHER THE DIMENSION 'NS' IS BIG ENOUGH
C
      XNS=AMAX1(1.,GD*BD,GD/(1.-BD))
      NXNS=2*(INT(XNS/2.)+1)
      IF(NS.LT.NXNS)WRITE(6,9923)NXNS
9923  FORMAT(/" * * * WARNING: THE INPUT 'NS' SHOULD BE AT LEAST",
     * I4,"  * * *"/)
C
        FDCPM=FD
68      DET0=0.
C
C     INITIALIZE RES. FREQ. OF PREVIOUS MODE AND DET. STORAGE VARIABLE
C
        DO 23 MODE=1,MODES
C
C     INIT GAP AND DETERMINE APPROPRIATE FREQ. FOR START OF SEARCH
C
          GAPP=GAP*E
          FDCREF=FD
          IF(MODE.GT.1)FDCREF=FDCPM
          IF((MODE.EQ.1).AND.(KMM.GT.1).AND.(GDG.LT.0))FDCREF
     *    =FDCPGD
          FDC=FDCREF-(GAPP/E)
          MMI=MM
          II=0
          IF(GG.NE.'Y')GO TO 17
          WRITE(6,900)
900       FORMAT(/8X,"FD/C",13X,"DET",16X,"GAP",15X,"SLOPE"/)
```

```
C
C     LOOP THROUGH ITERATIONS CLOSING IN ON THE RES. FREQ.
C
17        DO 13 KK=1,MMI
            IF(K.EQ.1)GO TO 18
            GAPP=ABS(GAPP)/E
15          FDC=FDC+GAPP
            II=II+1
C
C     CHECK IF ITERATION COUNT HAS BEEN EXCEEDED; CNE = ETA
C
            IF((K.EQ.0).AND.(II.GT.MAXSTEP))GO TO 777
18          HE=BD*(FDC**1.5)/SQRT(240.0E+06*PI)
            CNE=CMPLX(HE,HE)
            IS=0
            JS=0
C
C     CALCULATE (U-ALPHA*S*N*I*INVERSE(U-ETA*I)*N)
C     THE SUM ALPHA*N*I*INVERSE(U-ETA*I)*N  IS TREATED AS AN
C     INTEGRAL AND CALCULATED USING "CGAUSS"
C     THE RESULT IS MULTIPLIED BY 2 TO COVER THE WHOLE
C     RANGE OF INTEGRATION FROM MINUS INFINITY TO INFINITY
C
            DO 2 I=1,NS
            IF(IS.EQ.2)IS=0
            S1=FLOAT(I-1)
            RSS=SS(S1,FDC,BD,GD,CNE)
            DO 1 J=1,NS
            IF(JS.EQ.2)JS=0
            S2=FLOAT(J-1)
            W(I,J)=(0.,0.)
            IF((IS.EQ.0).AND.(JS.EQ.1))GO TO 1
            IF((IS.EQ.1).AND.(JS.EQ.0))GO TO 1
            A=CGAUSS(F,X1,X2,ERROR)
            ERROR1=ERROR*10.
            A=CGAUSS(F,X2,XM,ERROR1)+A
            W(I,J)=RSS*A*2.
1           JS=JS+1
2           IS=IS+1
C
C     NOW THAT ALL THE MATRIX ELEMENTS HAVE BEEN FOUND FINISH
C     CALCULATING THE MATRIX BY SUBTRACTING FROM THE UNIT MATRIX
C
            DO 6 I=1,NS
6           W(I,I)=1.-W(I,I)
C
C     USE "CMLIN" TO CALCULATE THE DET. OF THE MATRIX AND IF K=1
C     ALSO THE INVERSE OF THE MATRIX FOR USE IN OBTAINING Q AND Z
C
            CALL CMLIN(W,WW,20,NS,0,DET,V,K)
            IF(K.EQ.1)GO TO 14
            DT=REAL(DET)
            SL=(DT-DET0)/GAPP
            IF(GG.NE.'Y')GO TO 889
            WRITE(6,888)FDC,DT,GAPP,SL
888         FORMAT(5X,F9.6,7X,E10.4,7X,F12.9,7X,E12.5)
C
C     ON THE FIRST PASS THERE IS NO OLD DETERMINANT VALUE TO COMPARE
C     AGAINST SO CURRENT VALUES ARE JUST STORED.  ON SECOND PASS
C     ONLY THE DETERMINANT IS LOOKED AT AND A VALID SLOPE VALUE IS
C     STORED.  IN GENERAL, THERE IS FIRST A CHECK FOR A CHANGE IN
C     THE SIGN OF THE DETERMINANT.  IF ONE IS FOUND CONVERGENCE IS
C     BEGUN, OTHERWISE CHANGES IN THE SIGN OF THE SLOPE ARE SAUGHT.
C     THERE ARE FOUR CASES OF SLOPE AND DETERMINANT SIGN.  FOR THE
```

```
C     CASES +,+ AND -,- RESPECTIVELY,  THE CHANGE IS IGNORED TO
C     AVOID CONVERGENCE TO A LOCAL MAX OR MIN, RESPECTIVELY.
C     FOR THE +,- AND -,+ CASES, CONVERGENCE IS ATTEMPTED AND THE
C     CURRENT VALUES NECESSARY FOR A RESTART OF THE SEARCH ARE
C     STORED IN CASE OF A FALSE CONVERGENCE.
C
889         IF(II.EQ.1)GO TO 12
            IF((DT/DETO).GT.0.)GO TO 891
890         FDC=FDC-GAPP
            GO TO 13
891         IF(II.EQ.2)GO TO 12
            IF((SL/SLPO).GT.0.)GO TO 12
            IF((DETO/SLPO).GT.0.)GO TO 12
            IF(FCFLAG.EQ.1)GO TO 890
            SMMI=MMI-KK+1
            SGAP=GAPP
            FCFLAG=1
            GO TO 890
12          DETO=DT
            SLPO=SL
            GO TO 15
C
C     IN THE FOLLOWING CODE, THE RIGHT HAND SIDE OF THE EQUATION
C     IS CALCULATED FOR USE IN FINDING Q AND Z OF THE CAVITY
C     SUM = N*INVERSE(U-ALPHA*R*N*I*INVERSE(U-ETA*I)*N)*N*S
C
14          BETA=SQRT(1.-1./(GAMMA*GAMMA))
            XO=GD*FDC/(2.*BETA)
            SUM=(0.,0.)
            IS=0
            DO 22 I=1,NS
            IF(IS.EQ.2)IS=0
            S1=FLOAT(I-1)
            V(I)=XNPS(XO,S1,IS)
22          IS=IS+1
            DO 21 I=1,NS
            DO 21 J=1,NS
            S2=FLOAT(J-1)
C
C     HERE W(I,J) HAS BECOME THE MATRIX ELEMENT OF THE INVERSE MATRIX
C     AND UNI = INVERSE(U-ETA*I)
C
21          SUM=SUM+V(I)*W(I,J)*V(J)*SS(S2,FDC,BD,GD,CNE)
            UNI=1./(1.-CNE*RI(XO,FDC,BD,GD))
            SUM=SUM*UNI*UNI
            XZ=60.*GD/(BD*BD*FDC)
            Z=CMPLX(0.,-XZ)
            CHIB=FDC*BD/(BETA*GAMMA)
            CHIA=CHIB*AB
            Z=Z*SUM*2.*BESI1(CHIA)/(CHIA*BESIO(CHIB))
            ZR(KK)=REAL(Z)
            WV(KK)=FDC-FFDC
            FDC=FDC+GAPP
13          CONTINUE
C
C     CHECK FOR CONVERGENCE FOR CASE WHERE CONVERGENCE IS ATTEMPTED
C     BASED ON SLOPE CHANGES ONLY.  IF CONDITION IS NOT MET, ASSUME
C     CONVERGENCE IS FALSE AND RESTORE VALUES TO THOSE PRIOR TO
C     CONVERGENCE ATTEMPT AND BEGIN SEARCH BEYOND THIS "FALSE RES."
C
C     FCFLAG=0 IMPLIES TERMINATION OCCURS WHEN DET CHANGES SIGN.
C     FCFLAG=1 IMPLIES TERMINATION OCCURS WHEN SLOPE CHANGES SIGN.
C
        IF((FCFLAG.EQ.0).OR.((FCFLAG.EQ.1).AND.(ABS(DT).LT.EPSILON)))
```

```
     *    GO TO 39
C
C     SLOPE CHANGES SIGN BUT DET ITSELF IS NOT SMALL ENOUGH.  THIS
C     IS A FALSE CONVERGENCE.  SO WE NEED TO CONTINUE LOOP THAT
C     ENDS WITH STATEMENT 13.  NEW VALUES OF NN IS THEREFORE NEEDED
C     TO ENSURE A CONTINUATION.
C
          MMI=SMMI
          FDC=FDC+GAPP
          GAPP=SGAP
          FDC=FDC-GAPP
          GAPP=GAPP*E
          FCFLAG=0
          II=0
          GO TO 17
39        IF(K.EQ.1)GO TO 8
          FFDC=FDC+GAPP
          FDC=FFDC
          FDCPM=FFDC
          IF(MODE.EQ.1)FDCPGD=FFDC
          FCFLAG=0
          IF((MODE.EQ.1).AND.(GQ.EQ.'Y'))GO TO 41
          IF((GG.NE.'Y').AND.(KMM.GT.1))GO TO 43
          IF(MODE.GT.1)GO TO 43
          WRITE(6,42)
42        FORMAT(///7X,"G/D",4X,"FD/C",30X,"MODE"/)
43        WRITE(6,44)GD,FFDC,MODE
44        FORMAT(F10.3,F10.5,29X,I2//)
          GO TO 23
C
C     CALCULATION OF ACTUAL IMPEDANCE AND QUALITY:  PARAMETERS ARE
C     FIRST INITIALIZED TO FIND POINTS FOR LEAST SQUARES FIT TO Q
C
41        M=(KJ+1)/2
          FDC=FDC-.00001*FLOAT(M-1)
          K=1
          GAPP=.00001
          MMI=KJ
          GO TO 17
C
C     AFTER POINTS HAVE BEEN CALCULATED, CALCULATE VALUES NECCESSARY
C     FOR LEAST SQUARES FIT USING ADMITTANCE INSTEAD OF IMPEDANCE
C
8         S2=0.
          S2A=0.
          S4=0.
          DO 5 I=1,KJ
          WV2=WV(I)*WV(I)
          S2=S2+WV2
          S2A=S2A+WV2/ZR(I)
5         S4=S4+WV2*WV2
          TSUM=(ZR(M)*S2A-S2)/(120.*PI*S4)
C
C     THIS IS AN ARTIFACT OF CALCULATIONS FOR MODE>1
C     TSUM SOMETIMES IS NEGATIVE AS THE FUNCTIONAL
C     BEHAVIOR IS NOT AS EXPECTED
C
          IF(TSUM.GE.0.)GO TO 653
          WRITE(6,654)
654       FORMAT(/" * * *EITHER: MISMATCH OF EXTREMUM  OF IM(DET)* * *"
     *           /" * * *          AND ZERO OF RE(DET) DUE TO ERROR* * *"
     *           /" * * *OR:       CONVERGENCE OF A FALSE RESONANCE* * *")
          TSUM=ABS(TSUM)
653       Q=FFDC*0.5*SQRT(TSUM)*.001
```

```
          ZM=ZR(M)/(1000.*(120.*PI)**1.5)
          IF((GG.NE.'Y').AND.((KMM.GT.1).OR.(MODE.GT.1)))GO TO 750
          WRITE(6,300)
300       FORMAT(///7X,"G/D",4X,"FD/C",8X,"Z AT RES",8X,"Q"
     *     ,5X,"MODE"/)
750       WRITE(6,800)GD,FFDC,ZM,Q,MODE
800       FORMAT(F10.3,F10.5,E15.5,F10.4,4X,I2//)
23        K=0
25      GD=GD+GDG
        GO TO 9
777     WRITE(6,778)
778     FORMAT("//LOOP EXECUTED TOO MANY TIMES - RUN HALTED//")
        GO TO 9
7       STOP
        END
C
C       ***************************************************************
C
        COMPLEX FUNCTION SS(S,FDC,BD,GD,CNE)
        COMPLEX SSKI,CNE
C
C       THIS SUBROUTINE CALCULATES VALUES OF S.  SEE MATHEMATICAL
C       DEVELOPMENT FOR ACTUAL FORMULAS USED.  THE VARIOUS CASES
C       CONSIDERED ARE GAMMA IMAGINARY, REAL AND SMALL, AND REAL
C       AND LARGE. (MORE DETAIL IN DEVELOPMENT DOCUMENTATION)
C
        PI=3.1415927
        PSDG=PI*S/GD
        B=PSDG*PSDG-FDC*FDC
        CHID=SQRT(ABS(B))
        CHIB=CHID*BD
        IF(B.LT.0.)GO TO 3
        IF(CHID.GT.100..OR.CHIB.GT.100.)GO TO 5
        DI0=BESI0(CHID)
        DI1=BESI1(CHID)
        DK0=BESK0(CHID)
        DK1=BESK1(CHID)
        BI0=BESI0(CHIB)
        BI1=BESI1(CHIB)
        BK0=BESK0(CHIB)
        BK1=BESK1(CHIB)
        SSKI=CHIB*(DK0*BI0-BK0*DI0+CNE*(DK1*BI0+DI1*BK0)/CHIB)
     *          /(DK0*BI1+BK1*DI0-CNE*(-DK1*BI1+DI1*BK1)/CHIB)
        GO TO 4
3       DJ0=BESJ0(CHID)
        DJ1=BESJ1(CHID)
        DY0=BESY0(CHID)
        DY1=BESY1(CHID)
        BJ0=BESJ0(CHIB)
        BJ1=BESJ1(CHIB)
        BY0=BESY0(CHIB)
        BY1=BESY1(CHIB)
        SSKI=CHIB*(BJ0*DY0-DJ0*BY0-CNE*(DY1*BJ0-DJ1*BY0)/CHIB)
     *          /(BJ1*DY0-DJ0*BY1-CNE*(DY1*BJ1-DJ1*BY1)/CHIB)
        GO TO 4
5       CB=1./CHIB
        CD=1./CHID
        IF(CHIB.LE.90.)GO TO 7
        DI0=1.+.125*CD+.0703125*CD*CD
        DI1=1.-.375*CD-.1171875*CD*CD
        DK0=1.-.125*CD+.0703125*CD*CD
        DK1=1.+.375*CD-.1171875*CD*CD
        BI0=1.+.125*CB+.0703125*CB*CB
        BI1=1.-.375*CB-.1171875*CB*CB
```

```
      BK0=1.-.125*CB+.0703125*CB*CB
      BK1=1.+.375*CB-.1171875*CB*CB
      CHDB=CHID-CHIB
      IF(CHDB.GT.10.)GO TO 6
      EX=EXP(2.*CHDB)
      SSKI=CHIB*(DK0*BI0-EX*BK0*DI0+CNE*(DK1*BI0+EX*DI1*BK0)/CHIB)
     *       /(DK0*BI1+EX*BK1*DI0-CNE*(-DK1*BI1+EX*DI1*BK1)/CHIB)
      GO TO 4
6     SSKI=CHIB*(BK0/BK1)*(-DI0+CNE*DI1/CHIB)/
     *                  (DI0-CNE*DI1/CHIB)
      GO TO 4
7     SSKI=CHIB*(BESK0(CHIB)/BESK1(CHIB))*(-DI0+CNE*DI1/CHIB)/
     *                                    (DI0-CNE*DI1/CHIB)
4     B2=2.
      IF(S.LT..01)B2=1.
      SS=B2*SSKI-CNE*B2
      RETURN
      END
C
C     ******************************************************************
C
      COMPLEX FUNCTION F(X)
      COMPLEX X,CNE
      COMMON PI,IS,JS,S1,S2,FDC,BD,GD,AB,CNE
      RX=REAL(X)
C
C     THIS FUNCTION PROVIDES VALUES TO "CGAUSS" FOR THE INTEGRAL
C     BEING CALCULATE IN THE MAIN PROGRAM (SEE ADDITIONAL DOCUMENTATION)
C
      A=RI(RX,FDC,BD,GD)
      XN=XNPS(RX,S1,IS)*XNPS(RX,S2,JS)/PI
      F=XN*A/(1.-CNE*A)
      RETURN
      END
C
C     ******************************************************************
C
      FUNCTION RI(X,FDC,BD,GD)
C
C     THIS FUNCTION RETURNS VALUES FOR I, A RATIO OF BESSEL FUNCTIONS
C     TIMES THE INVERSE OF (CHI*B).  THE CASES CONSIDERED ARE DISCUSSED
C     IN THE ADDITIONAL DOCUMENTATION
C
      A=2.*X/GD
      B=A*A-FDC*FDC
      CHI=SQRT(ABS(B))*BD
      IF(B.LT.0.)GO TO 1
      IF(CHI.LT..01)GO TO 2
      IF(CHI.GT.100.)GO TO 4
      RI=BESI1(CHI)/(CHI*BESI0(CHI))
      RETURN
1     IF(CHI.LT..01)GO TO 3
      RI=BESJ1(CHI)/(CHI*BESJ0(CHI))
      RETURN
2     RI=0.5*(1.-CHI*CHI/8.+CHI*CHI*CHI*CHI/48.)
      RETURN
3     RI=0.5*(1.+CHI*CHI/8.+CHI*CHI*CHI*CHI/48.)
      RETURN
4     C=1./CHI
      RI=C*(1.+.5*C+.375*C*C)
      RETURN
      END
C
C     ******************************************************************
```

```
C
      FUNCTION XNPS(X,S,I)
C
C     THIS FUNCTION RETURNS ELEMENTS OF THE MATRIX N
C     THE CASES CONSIDER EVEN AND ODD INDICES AND
C     APPROXIMATIONS DEPENDANT ON THE INPUTS
C
      PI=3.1415927
      CS=PI*S/2.
      IF(ABS(X-CS).LT..05)GO TO 1
      A=X/(X*X-CS*CS)
      IF(I.EQ.0)XNPS=A*SIN(X)
      IF(I.EQ.1)XNPS=A*COS(X)
      RETURN
1     A=X*(1.-(X-CS)*(X-CS)/6.)/(X+CS)
      IF(I.EQ.1)GO TO 2
      K=INT(S/2.+.01)
      IF(MOD(K,2).EQ.0)XNPS=A
      IF(MOD(K,2).EQ.1)XNPS=-A
      RETURN
2     K=INT((S+1.)/2.+0.01)
      IF(MOD(K,2).EQ.0)XNPS=A
      IF(MOD(K,2).EQ.1)XNPS=-A
      RETURN
      END
```

APPENDIX II

The following is the batch job deck TESTCAV for running the program CAVITY. The input file is LFN1 and the output file is LFN2. Needless to say, the user's number, password and charge number should be updated.

```
TESTCAV.
USER,99999,99999.
CHARGE,999.
AT,CERNLIB/UN=LIBR.
LIBRARY(CERNLIB).
G,LFN1.
AT,CAV/UN=94786.
CAV,LFN1,LFN2.
REPLACE,LFN2.
DAYFILE,DAYFIL.
REPLACE,DAYFIL.
EXIT.
DAYFILE,DAYFIL.
REPLACE,DAYFIL.
```

APPENDIX III

The following is a listing of the program CAVPLT written using ANSI '77 FORTRAN in CDC FORTRAN 5. It is available from a Fermilab terminal by entering GET,CAVPLT/UN=94786.

```
      PROGRAM CAVPLT(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,DIGF,
     *TAPE99=DIGF)
      EXTERNAL F
      COMMON PI,IS,JS,S1,S2,FDC,BD,GD,CNE
      COMPLEX A,X1,X2,XM,F,SUM,Z,UNI,DET,RSS
      COMPLEX CNE,SS,CGAUSS,W(20,20),WW(20)
      CHARACTER GG*1,GQ*1
      INTEGER PPC,CPG
      DIMENSION V(20),ZR(31),WV(31),IFLAG(45),PARA(22),ZX(2),ZY(2)
      DIMENSION FREQM(300),RDETM(300),ZDETM(300)
      NAMELIST/PLOT/BD,GD,FDST,FDEND,GAP,GG,XI,XMAX,ERROR,NS
      PI=3.1415927
      E=2.71828
      GG='R'
      FDST=2.405
      XI=10.
      XMAX=100.
      NS=6
      ERROR=.00001
      READ(5,PLOT,END=51)
      IF((BD.LE.0.).OR.(BD.GT.1.))GO TO 892
      IF(GD.LE.0.)GO TO 892
      IF(FDST.LT.2.40)GO TO 892
      NPOINT=INT((FDEND-FDST)/GAP)+1
      IF(NPOINT.GT.300)THEN
      WRITE(6,987)NPOINT
987   FORMAT(/" * * * NO. OF POINTS IS",I4," WHICH IS BIGGER ",
     *  "THAN 300 * * *"/
     *        "       EITHER DECREASE NO. OF POINTS OR CHANGE ",
     *  "DIMENSION STATEMENT"//)
      GO TO 51
      ELSE
      GO TO 9
      END IF
9     IF((GAP.GT.0.).AND.(GAP.LE.1.))GO TO 10
892   WRITE(6,893)
893   FORMAT(/" * * * ERROR IN INPUT PARAMETERS * * *")
      GO TO 51
10    X1=(0.,0.)
      X2=CMPLX(XI,0.)
      XM=CMPLX(XMAX,0.)
      GAPP=GAP
      FDC=FDST-GAPP
      K=0
      II=0
15    FDC=FDC+GAPP
      II=II+1
18    HE=BD*(FDC**1.5)/SQRT(240.0E+06*PI)
      CNE=CMPLX(HE,HE)
      IS=0
```

```
      JS=0
      DO 2 I=1,NS
      IF(IS.EQ.2)IS=0
      S1=FLOAT(I-1)
      RSS=SS(S1,FDC,BD,GD,CNE)
      DO 1 J=1,NS
      IF(JS.EQ.2)JS=0
      S2=FLOAT(J-1)
      W(I,J)=(0.,0.)
      IF((IS.EQ.0).AND.(JS.EQ.1))GO TO 1
      IF((IS.EQ.1).AND.(JS.EQ.0))GO TO 1
      A=CGAUSS(F,X1,X2,ERROR)
      ERROR1=ERROR*10.
      A=CGAUSS(F,X2,XM,ERROR1)+A
      W(I,J)=RSS*A*2.
1     JS=JS+1
2     IS=IS+1
      DO 6 I=1,NS
6     W(I,I)=1.-W(I,I)
      CALL CMLIN(W,WW,20,NS,0,DET,V,K)
      DT=REAL(DET)
      AI=AIMAG(DET)
      FREQM(II+2)=FDC
      RDETM(II+2)=DT
      ZDETM(II+2)=AI
      PPC=II+2
      PRINT *,II,FDC,DT,AI
      IF(FDC.GT.FDEND)GO TO 7
      GO TO 15
7     RDETM(1)=0.
      RDETM(2)=0.
      ZDETM(1)=0.
      ZDETM(2)=0.
      FREQM(1)=FDEND
      FREQM(2)=FDST
      DO 40 I=1,45
40    IFLAG(I)=0
      IFLAG(1)=1
      PARA(1)=PPC
      IFLAG(3)=1
      PARA(3)=-1
      IFLAG(37)=1
      IFLAG(45)=0
      IF(GG.NE.'R')GO TO 53
      CALL PLTDSP(' ',0,' ',0,' ',0,FREQM,RDETM,DUMMY,PARA,IFLAG)
      GO TO 51
53    CALL PLTDSP(' ',0,' ',0,' ',0,FREQM,ZDETM,DUMMY,PARA,IFLAG)
51    STOP
      END
      COMPLEX FUNCTION SS(S,FDC,BD,GD,CNE)
      COMPLEX SSKI,CNE
      PI=3.1415927
      PSDG=PI*S/GD
      B=PSDG*PSDG-FDC*FDC
      CHID=SQRT(ABS(B))
      CHIB=CHID*BD
      IF(B.LT.0.)GO TO 3
      IF(CHID.GT.100..OR.CHIB.GT.100.)GO TO 5
      DI0=BESI0(CHID)
      DI1=BESI1(CHID)
      DK0=BESK0(CHID)
      DK1=BESK1(CHID)
      BI0=BESI0(CHIB)
      BI1=BESI1(CHIB)
```

```
      BK0=BESK0(CHIB)
      BK1=BESK1(CHIB)
      SSKI=CHIB*(DK0*BI0-BK0*DI0+CNE*(DK1*BI0+DI1*BK0)/CHIB)
     *          /(DK0*BI1+BK1*DI0-CNE*(-DK1*BI1+DI1*BK1)/CHIB)
      GO TO 4
3     DJ0=BESJ0(CHID)
      DJ1=BESJ1(CHID)
      DY0=BESY0(CHID)
      DY1=BESY1(CHID)
      BJ0=BESJ0(CHIB)
      BJ1=BESJ1(CHIB)
      BY0=BESY0(CHIB)
      BY1=BESY1(CHIB)
      SSKI=CHIB*(BJ0*DY0-DJ0*BY0-CNE*(DY1*BJ0-DJ1*BY0)/CHIB)
     *          /(BJ1*DY0-DJ0*BY1-CNE*(DY1*BJ1-DJ1*BY1)/CHIB)
      GO TO 4
5     CB=1./CHIB
      CD=1./CHID
      IF(CHIB.LE.90.)GO TO 7
      DI0=1.+.125*CD+.0703125*CD*CD
      DI1=1.-.375*CD-.1171875*CD*CD
      DK0=1.-.125*CD+.0703125*CD*CD
      DK1=1.+.375*CD-.1171875*CD*CD
      BI0=1.+.125*CB+.0703125*CB*CB
      BI1=1.-.375*CB-.1171875*CB*CB
      BK0=1.-.125*CB+.0703125*CB*CB
      BK1=1.+.375*CB-.1171875*CB*CB
      CHDB=CHID-CHIB
      IF(CHDB.GT.10.)GO TO 6
      EX=EXP(2.*CHDB)
      SSKI=CHIB*(DK0*BI0-EX*BK0*DI0+CNE*(DK1*BI0+EX*DI1*BK0)/CHIB)
     *        /(DK0*BI1+EX*BK1*DI0-CNE*(-DK1*BI1+EX*DI1*BK1)/CHIB)
      GO TO 4
6     SSKI=CHIB*(BK0/BK1)*(-DI0+CNE*DI1/CHIB)/
     *                    (DI0-CNE*DI1/CHIB)
      GO TO 4
7     SSKI=CHIB*(BESK0(CHIB)/BESK1(CHIB))*(-DI0+CNE*DI1/CHIB)/
     *                                     (DI0-CNE*DI1/CHIB)
4     B2=2.
      IF(S.LT..01)B2=1.
      SS=B2*SSKI-CNE*B2
      RETURN
      END
      COMPLEX FUNCTION F(X)
      COMPLEX X,CNE
      COMMON PI,IS,JS,S1,S2,FDC,BD,GD,CNE
      RX=REAL(X)
      A=RI(RX,FDC,BD,GD)
      XN=XNPS(RX,S1,IS)*XNPS(RX,S2,JS)/PI
      F=XN*A/(1.-CNE*A)
      RETURN
      END
      FUNCTION RI(X,FDC,BD,GD)
      A=2.*X/GD
      B=A*A-FDC*FDC
      CHI=SQRT(ABS(B))*BD
      IF(B.LT.0.)GO TO 1
      IF(CHI.LT..01)GO TO 2
      IF(CHI.GT.100.)GO TO 4
      RI=BESI1(CHI)/(CHI*BESI0(CHI))
      RETURN
1     IF(CHI.LT..01)GO TO 3
      RI=BESJ1(CHI)/(CHI*BESJ0(CHI))
      RETURN
```

```
2     RI=0.5*(1.-CHI*CHI/8.+CHI*CHI*CHI*CHI/48.)
      RETURN
3     RI=0.5*(1.+CHI*CHI/8.+CHI*CHI*CHI*CHI/48.)
      RETURN
4     C=1./CHI
      RI=C*(1.+.5*C+.375*C*C)
      RETURN
      END
      FUNCTION XNPS(X,S,I)
      PI=3.1415927
      CS=PI*S/2.
      IF(ABS(X-CS).LT..05)GO TO 1
      A=X/(X*X-CS*CS)
      IF(I.EQ.0)XNPS=A*SIN(X)
      IF(I.EQ.1)XNPS=A*COS(X)
      RETURN
1     A=X*(1.-(X-CS)*(X-CS)/6.)/(X+CS)
      IF(I.EQ.1)GO TO 2
      K=INT(S/2.+.01)
      IF(MOD(K,2).EQ.0)XNPS=A
      IF(MOD(K,2).EQ.1)XNPS=-A
      RETURN
2     K=INT((S+1.)/2.+0.01)
      IF(MOD(K,2).EQ.0)XNPS=A
      IF(MOD(K,2).EQ.1)XNPS=-A
      RETURN
      END
```

APPENDIX IV

The following is a listing of the batch job deck TESTP for executing the program CAVPLT in order to create a digfile DIGFP for plotting Re(detW). The input file is named LFN. Needless to say, the user's number, password and charge number should be updated.

```
TESTP.
USER,99999,99999.
CHARGE,999.
SETTL(*).
PURGE,OUTP/NA.
DEFINE,OUTPUT=OUTP.
AT,CERNLIB/UN=LIBR.
AT,GRALIB/UN=LIBR.
LIBRARY(CERNLIB,GRALIB).
G,LFN.
AT,CAVP/UN=94786.
CAVP,LFN.
REPLACE,DIGFP.
DAYFILE,DAYFILP.
REPLACE,DAYFILP.
EXIT.
DAYFILE,DAYFILP.
REPLACE,DAYFILP.
```

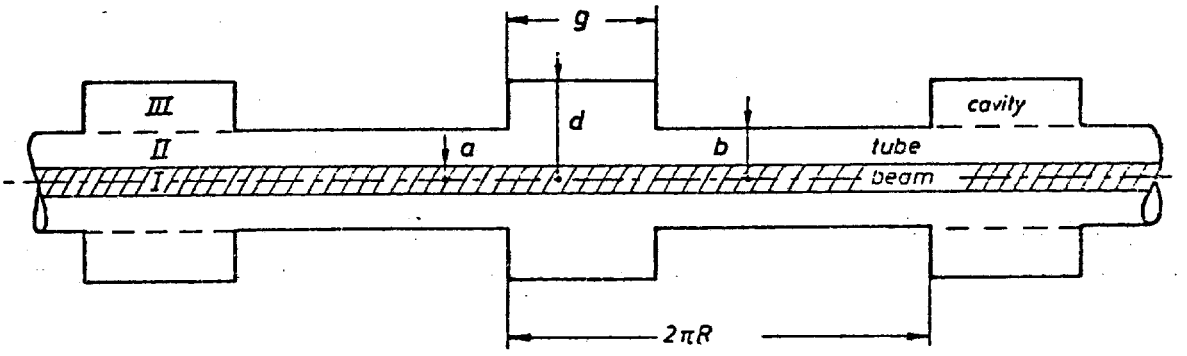


Figure 1. Schematic cross section of the corrugated beampipe.

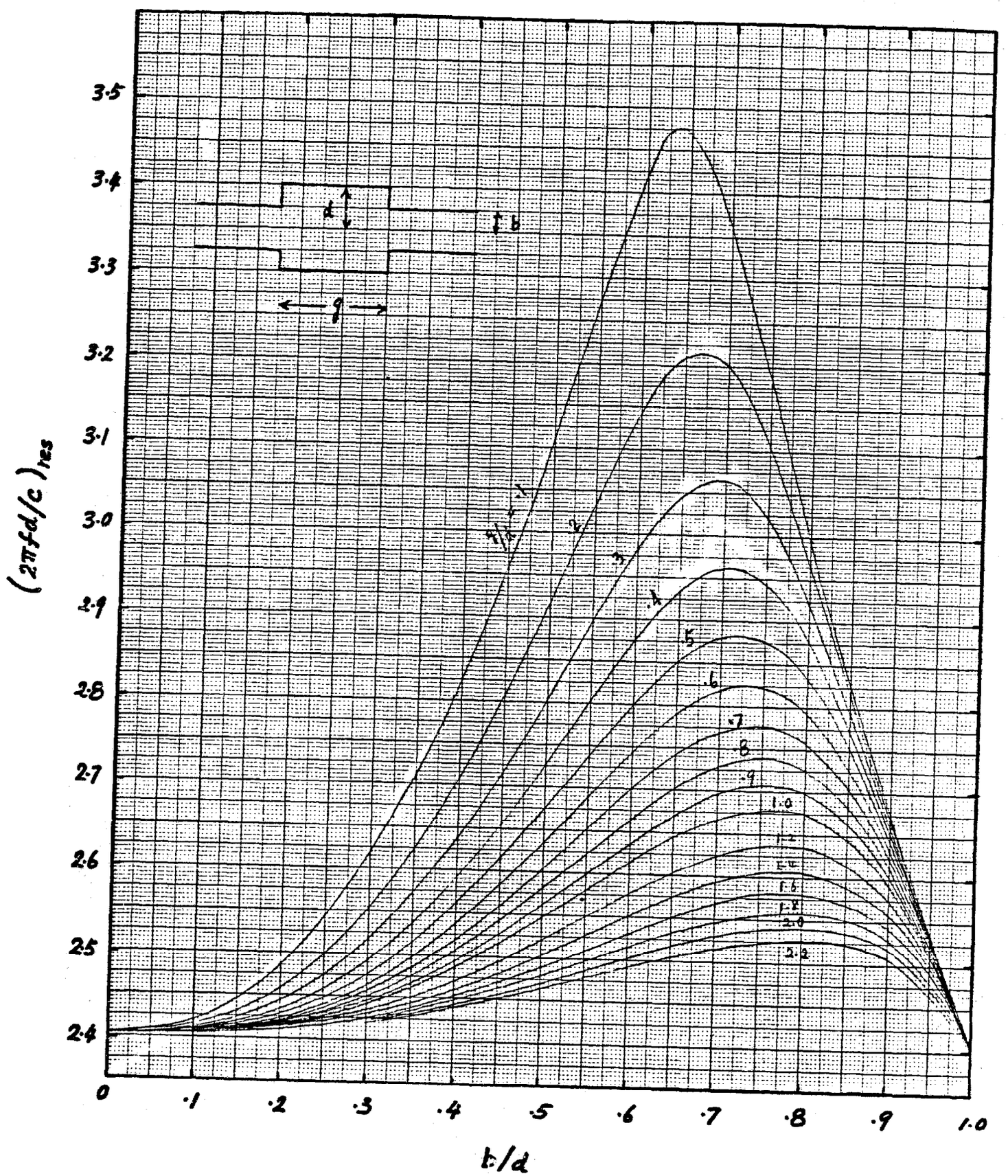
3.5
3.4
3.3
3.2
3.1
3.0
2.9
2.8
2.7
2.6
2.5
2.4
$(2\pi fd/c)_{res}$
d
b
g
0
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
b/d
1
2
3
4
.5
.6
.7
.8
.9
1.0
1.2
1.4
1.6
1.8
2.0
2.2

Figure 2

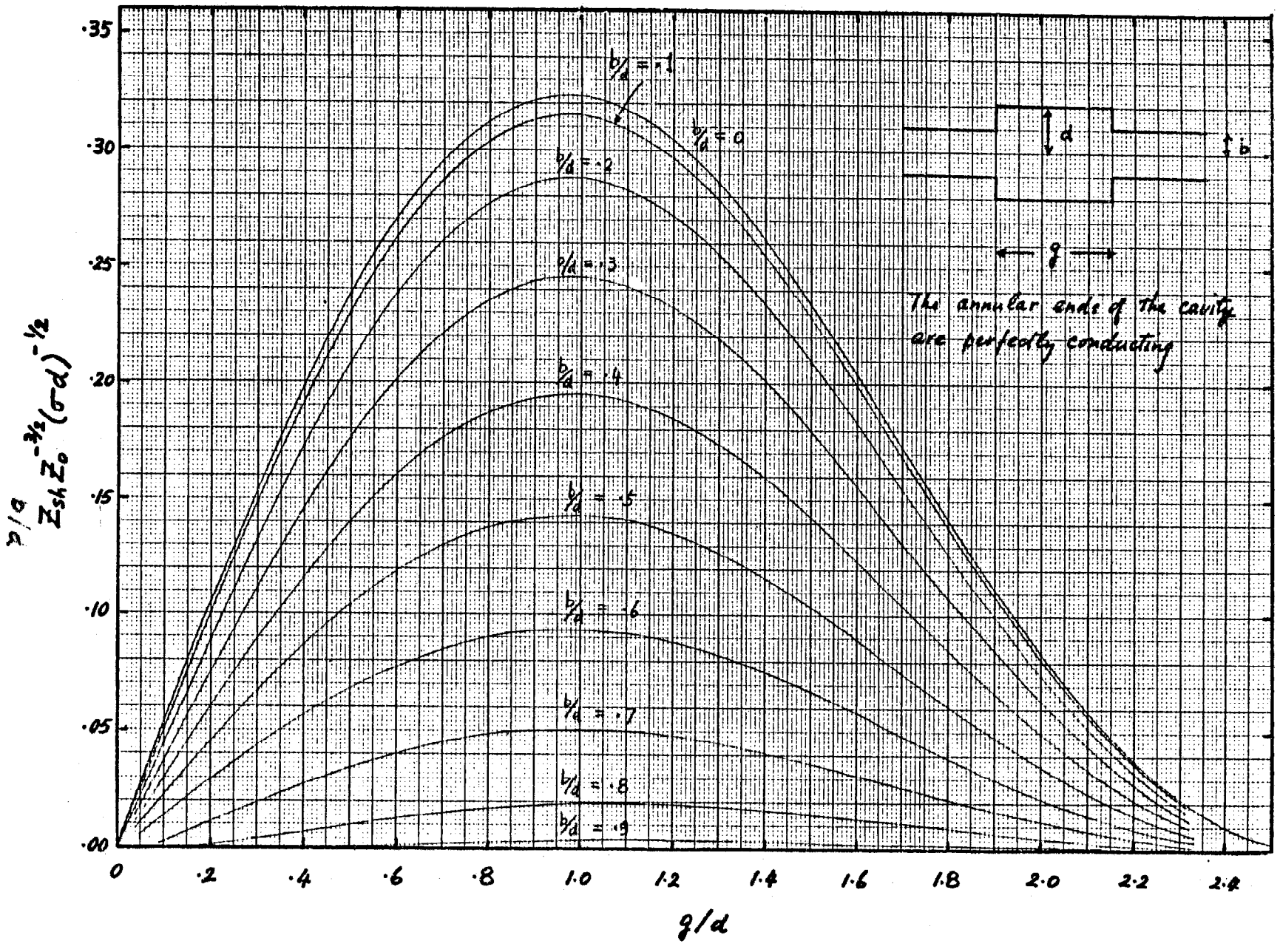
$b/a \, Z_{sh} Z_0^{-3/2} (\sigma d)^{-1/2}$
.35
.30
.25
.20
.15
.10
.05
.00
0
.2
.4
.6
.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
g/d
b/d = .1
b/d = 0
b/d = .2
b/d = .3
b/d = .4
b/d = .5
b/d = .6
b/d = .7
b/d = .8
b/d = .9
d
b
g
The annular ends of the cavity are perfectly conducting

Figure 3

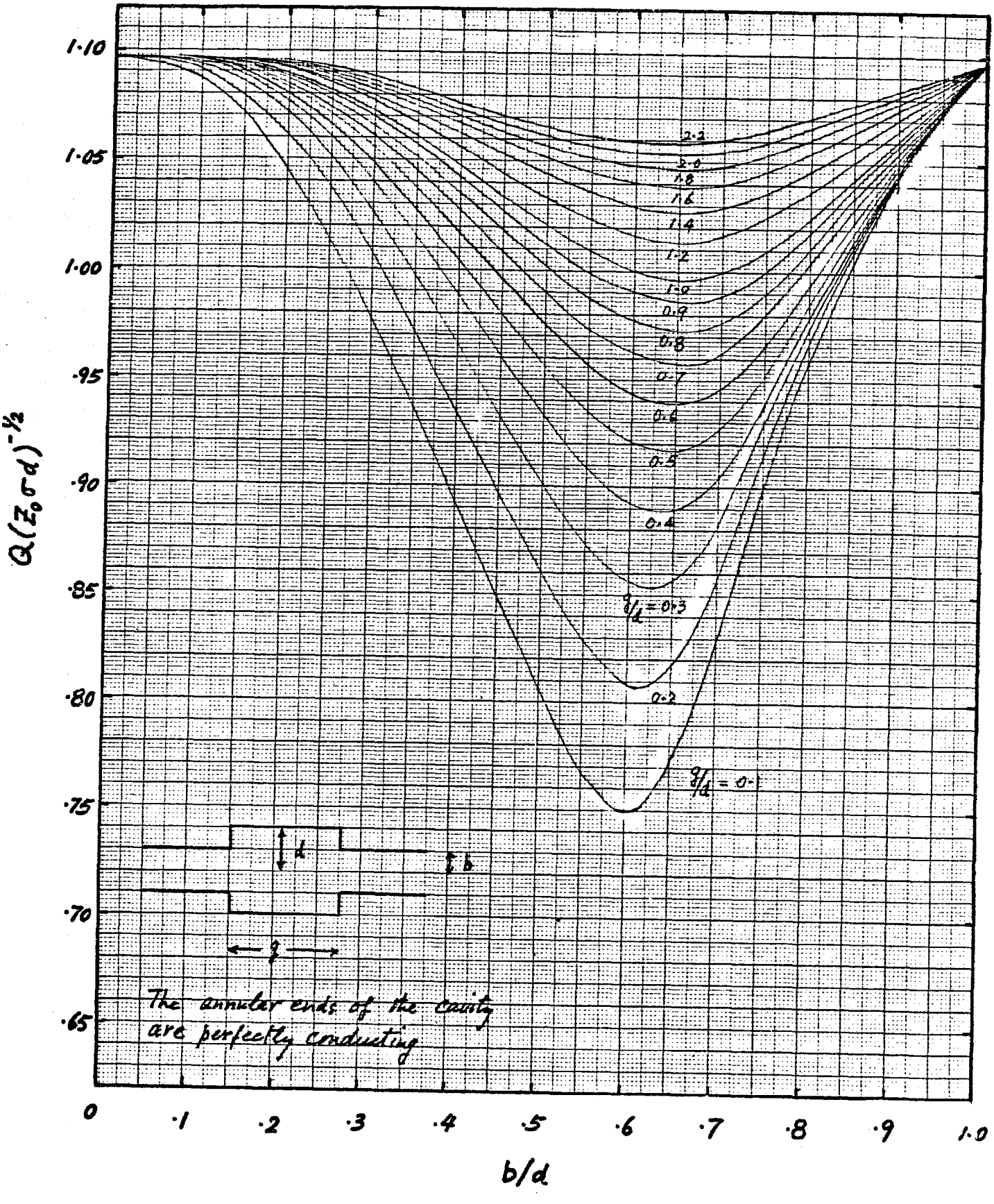
1·10
1·05
1·00
·95
·90
·85
·80
·75
·70
·65
$Q(Z_0 \sigma d)^{-1/2}$
2·2
2·0
1·8
1·6
1·4
1·2
1·0
0·9
0·8
0·7
0·6
0·5
0·4
$g/d = 0·3$
0·2
$g/d = 0·1$
d
b
g
The annular ends of the cavity are perfectly conducting
0
·1
·2
·3
·4
·5
·6
·7
·8
·9
1·0
b/d

Figure 4

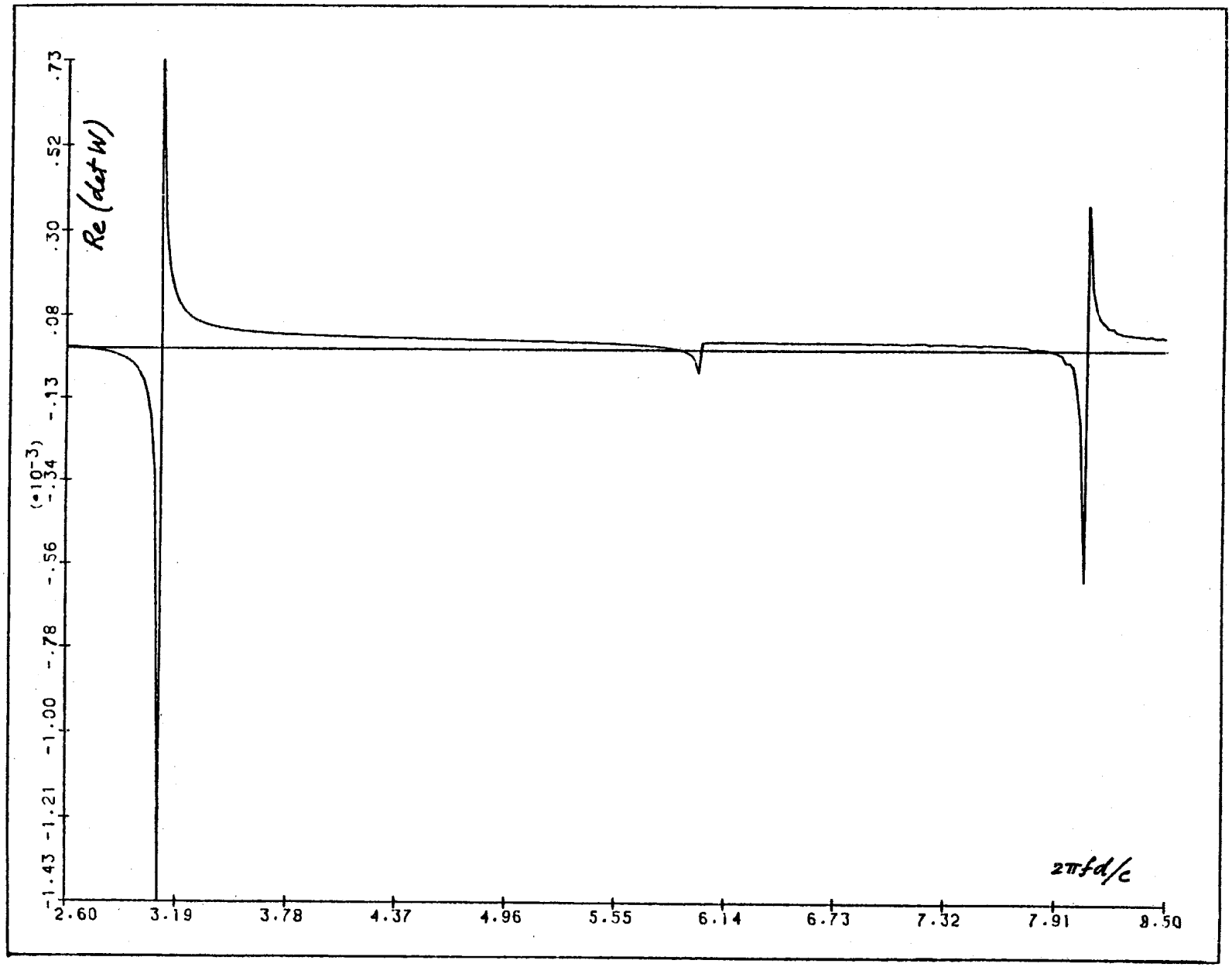
Re (det W)
(*10^-3)
.73
.52
.30
.08
-.13
-.34
-.56
-.78
-1.00
-1.21
-1.43
2.60
3.19
3.78
4.37
4.96
5.55
6.14
6.73
7.32
7.91
8.50
2πfd/c

Figure 5

Table 1. Output for $DATA BD=.5 GD=.8,MODES=3$.

NOTE: RESONANCE FREQUENCY F IS OBTAINED FROM FD/C BY USING:

FD/C = 2*PI*F*D/C,

WHERE C IS THE VELOCITY OF LIGHT AND D IS THE CAVITY RADIUS.

THE TRUE RESONANCE RESISTANCE IS

Z*SQRT(SIGMA*D)*Z0**1.5,

WHERE Z0 = 120*PI OHMS AND SIGMA IS THE WALL CONDUCTIVITY.

THE FIGURE OF MERIT IS

Q*SQRT(SIGMA*D*Z0).

B/D = .5000 A/B = .005000 GAMMA = 1000.0000
NS = 6 XMAX = 100.

| G/D | FD/C | Z AT RES | Q | MODE |
|---|---|---|---|---|
| .800 | 2.58092 | .13779E+00 | 1.0021 | 1 |
| .800 | 3.58790 | | | 2 |
| .800 | 4.68222 | | | 3 |

Table 2. Output for $DATA BD=.5,GD=1.,KMKM=8,GDG=-.1$.

NOTE: RESONANCE FREQUENCY F IS OBTAINED FROM FD/C BY USING:

FD/C = 2*PI*F*D/C, .

WHERE C IS THE VELOCITY OF LIGHT AND D IS THE CAVITY RADIUS.

THE TRUE RESONANCE RESISTANCE IS

Z*SQRT(SIGMA*D)*Z0**1.5,

WHERE Z0 = 120*PI OHMS AND SIGMA IS THE WALL CONDUCTIVITY.

THE FIGURE OF MERIT IS

Q*SQRT(SIGMA*D*Z0).

B/D = .5000 A/B = .005000 GAMMA = 1000.0000
NS = 6 XMAX = 100.

| G/D | FD/C | Z AT RES | Q | MODE |
|---|---|---|---|---|
| 1.000 | 2.54592 | .14312E+00 | 1.0204 | 1 |
| .900 | 2.56167 | .14223E+00 | 1.0121 | 1 |
| .800 | 2.58092 | .13779E+00 | 1.0021 | 1 |
| .700 | 2.60486 | .12979E+00 | .9900 | 1 |
| .600 | 2.63530 | .11835E+00 | .9750 | 1 |
| .500 | 2.67500 | .10370E+00 | .9562 | 1 |
| .400 | 2.72852 | .86219E-01 | .9323 | 1 |
| .300 | 2.80390 | .66394E-01 | .9014 | 1 |

Table 3. Output for $DATA BD=.5,GD=.8,GG='Y',GAP=.1$.

NOTE: RESONANCE FREQUENCY F IS OBTAINED FROM FD/C BY USING:

FD/C = 2*PI*F*D/C,

WHERE C IS THE VELOCITY OF LIGHT AND D IS THE CAVITY RADIUS.

THE TRUE RESONANCE RESISTANCE IS

Z*SQRT(SIGMA*D)*Z0**1.5,

WHERE Z0 = 120*PI OHMS AND SIGMA IS THE WALL CONDUCTIVITY.

THE FIGURE OF MERIT IS

Q*SQRT(SIGMA*D*Z0).

B/D = .5000 A/B = .005000 GAMMA = 1000.0000
NS = 6 XMAX = 100.

| FD/C | DET | GAP | SLOPE |
|---|---|---|---|
| 2.405000 | .3328E+01 | .100000000 | .33278E+02 |
| 2.505000 | .1568E+01 | .100000000 | -.17595E+02 |
| 2.605000 | -.5477E+00 | .100000000 | -.21160E+02 |
| 2.541788 | .8366E+00 | .036787969 | -.19890E+02 |
| 2.578576 | .5187E-01 | .036787969 | -.21331E+02 |
| 2.615364 | -.7917E+00 | .036787969 | -.22931E+02 |
| 2.592109 | -.2513E+00 | .013533547 | -.22400E+02 |
| 2.583555 | -.5871E-01 | .004978717 | -.22211E+02 |
| 2.580408 | .1132E-01 | .001831569 | -.22142E+02 |
| 2.582239 | -.2938E-01 | .001831569 | -.22222E+02 |
| 2.581081 | -.3639E-02 | .000673797 | -.22197E+02 |
| 2.580655 | .5817E-02 | .000247876 | -.22188E+02 |
| 2.580903 | .3148E-03 | .000247876 | -.22199E+02 |
| 2.581151 | -.5190E-02 | .000247876 | -.22209E+02 |
| 2.580994 | -.1710E-02 | .000091189 | -.22206E+02 |
| 2.580937 | -.4301E-03 | .000033546 | -.22205E+02 |
| 2.580916 | .4077E-04 | .000012341 | -.22204E+02 |
| 2.580928 | -.2333E-03 | .000012341 | -.22205E+02 |
| 2.580920 | -.6003E-04 | .000004540 | -.22205E+02 |
| 2.580917 | .3689E-05 | .000001670 | -.22205E+02 |
| 2.580919 | -.3340E-04 | .000001670 | -.22205E+02 |
| 2.580918 | -.9954E-05 | .000000614 | -.22205E+02 |
| 2.580917 | -.1330E-05 | .000000226 | -.22205E+02 |
| 2.580917 | .1843E-05 | .000000083 | -.22205E+02 |
| 2.580917 | -.3563E-08 | .000000083 | -.22205E+02 |

| G/D | FD/C | Z AT RES | Q | MODE |
|---|---|---|---|---|
| .800 | 2.58092 | .13779E+00 | 1.0021 | 1 |